python coding questions @hiringhustle

For Loop:

Here are some Python problem-solving interview questions related to `for` loops along with their answers:

1. **Iterate Over List:**
    - Question: Write a Python function to iterate over elements of a list using a `for` loop.
    - Answer:

    ```
    def iterate_list(lst):
        for item in lst:
            print(item)
    ```

2. **Iterate Over Dictionary:**

- Question: Write a Python function to iterate over key-value pairs of a dictionary using a `for` loop.
- Answer:

```
def iterate_dict(dictionary):
    for key, value in dictionary.items():
        print(key, value)
```

3. **Generate Fibonacci Series:**

- Question: Write a Python function to generate the Fibonacci series up to a specified limit using a `for` loop.
- Answer:

```
def fibonacci_series(limit):
    fib = [0, 1]
    for i in range(2, limit):
        fib.append(fib[-1] + fib[-2])
    return fib
```

4. **Count Even Numbers:**

- Question: Write a Python function to count the number of even numbers in a given list using a `for` loop.
- Answer:

```
def count_even_numbers(lst):
    count = 0
    for num in lst:
        if num % 2 == 0:
            count += 1
    return count
```

5. **Find Maximum Element:**

- Question: Write a Python function to find the maximum element in a given list using a `for` loop.
- Answer:

```python
def find_max(lst):
    max_value = float('-inf')
    for num in lst:
        if num > max_value:
            max_value = num
    return max_value
```

6. **Calculate Factorial:**
    - Question: Write a Python function to calculate the factorial of a given number using a `for` loop.
    - Answer:

```python
def factorial(n):
    result = 1
    for i in range(1, n+1):
        result *= i
    return result
```

7. **Reverse String:**
    - Question: Write a Python function to reverse a given string using a `for` loop.
    - Answer:

```python
def reverse_string(s):
    reversed_str = ''
    for i in range(len(s)-1, -1, -1):
        reversed_str += s[i]
    return reversed_str
```

8. **Find Prime Numbers:**
    - Question: Write a Python function to find all prime numbers up to a given limit using a `for` loop.
    - Answer:

```python
def find_primes(limit):
    primes = []
    for num in range(2, limit+1):
        is_prime = True
        for i in range(2, int(num**0.5)+1):
            if num % i == 0:
                is_prime = False
                break
        if is_prime:
            primes.append(num)
    return primes
```

These questions demonstrate various applications of `for` loops in Python, including iterating over lists and dictionaries, generating sequences, counting occurrences, finding maximum elements, calculating factorials, reversing strings, and finding prime numbers.

while Loop:

Here are some Python problem-solving interview questions related to `while` loops along with their answers:

1. **Calculate Factorial Using While Loop:**
    - Question: Write a Python function to calculate the factorial of a given number using a `while` loop.
    - Answer:

    ```python
    def factorial(n):
        result = 1
        while n > 1:
            result *= n
            n -= 1
        return result
    ```

2. **Find Sum of Digits Using While Loop:**

- Question: Write a Python function to calculate the sum of digits of a given number using a `while` loop.
- Answer:

  ```
  def sum_of_digits(n):
      total = 0
      while n > 0:
          total += n % 10
          n //= 10
      return total
  ```

3. **Find Prime Numbers Using While Loop:**
   - Question: Write a Python function to find all prime numbers up to a given limit using a `while` loop.
   - Answer:

     ```
     def find_primes(limit):
         primes = []
         num = 2
         while num <= limit:
             is_prime = True
             divisor = 2
             while divisor * divisor <= num:
                 if num % divisor == 0:
                     is_prime = False
                     break
                 divisor += 1
             if is_prime:
                 primes.append(num)
             num += 1
         return primes
     ```

4. **Reverse a Number Using While Loop:**
   - Question: Write a Python function to reverse a given number using a `while` loop.
   - Answer:

```
def reverse_number(n):
    reversed_num = 0
    while n > 0:
        digit = n % 10
        reversed_num = reversed_num * 10 + digit
        n //= 10
    return reversed_num
```

5. **Check Armstrong Number Using While Loop:**
   - Question: Write a Python function to check if a given number is an Armstrong number using a `while` loop.
   - Answer:

```
def is_armstrong(n):
    order = len(str(n))
    temp = n
    total = 0
    while temp > 0:
        digit = temp % 10
        total += digit ** order
        temp //= 10
    return n == total
```

6. **Generate Fibonacci Series Using While Loop:**
   - Question: Write a Python function to generate the Fibonacci series up to a specified limit using a `while` loop.
   - Answer:

```
def fibonacci_series(limit):
    fib = [0, 1]
    while fib[-1] + fib[-2] <= limit:
        fib.append(fib[-1] + fib[-2])
    return fib
```

These questions demonstrate various applications of `while` loops in Python, including calculating factorials, summing digits, finding prime numbers, reversing numbers, checking for Armstrong numbers, and generating Fibonacci series.

basic problems:

Here are some Python problem-solving interview questions along with their answers:

1. **Reverse a String:**
    - Question: Write a Python function to reverse a string.
    - Answer:

    ```
    def reverse_string(string):
        return string[::-1]
    ```

2. **Check for Palindrome:**
    - Question: Write a Python function to check if a given string is a palindrome.
    - Answer:

    ```
    def is_palindrome(string):
        return string == string[::-1]
    ```

3. **Find the Missing Number:**
    - Question: Given a list of numbers from 1 to N with one missing, find the missing number.
    - Answer:

    ```
    def find_missing_number(nums):
        n = len(nums) + 1
        total_sum = n * (n + 1) // 2
        return total_sum - sum(nums)
    ```

4. **Check for Anagrams:**

- Question: Write a Python function to check if two strings are anagrams of each other.
- Answer:

```
def are_anagrams(str1, str2):
    return sorted(str1) == sorted(str2)
```

5. **Find the Maximum Element in a List:**
    - Question: Write a Python function to find the maximum element in a given list.
    - Answer:

```
def find_max(lst):
    return max(lst)
```

6. **Count Words in a Sentence:**
    - Question: Write a Python function to count the number of words in a given sentence.
    - Answer:

```
def count_words(sentence):
    return len(sentence.split())
```

7. **Remove Duplicates from a List:**
    - Question: Write a Python function to remove duplicates from a given list while preserving the order.
    - Answer:

```
def remove_duplicates(lst):
    return list(dict.fromkeys(lst))
```

8. **Check for Prime Number:**
    - Question: Write a Python function to check if a given number is prime.
    - Answer:

```
def is_prime(num):
    if num <= 1:
        return False
    for i in range(2, int(num ** 0.5) + 1):
        if num % i == 0:
            return False
    return True
```

Here are some more Python problem-solving interview questions along with their answers:

1. **Find Common Elements in Lists:**
   - Question: Write a Python function to find common elements between two lists.
   - Answer:

   ```
   def find_common_elements(list1, list2):
       return list(set(list1) & set(list2))
   ```

2. **Calculate Factorial:**
   - Question: Write a Python function to calculate the factorial of a given number.
   - Answer:

   ```
   def factorial(n):
       if n == 0:
           return 1
   ```

```
    else:
        return n * factorial(n-1)
```

3. **Reverse a Linked List:**

- Question: Write a Python function to reverse a linked list.
- Answer:

```
class ListNode:
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

def reverse_linked_list(head):
    prev = None
    while head:
        temp = head.next
        head.next = prev
        prev = head
        head = temp
    return prev
```

4. **Check for Armstrong Number:**

- Question: Write a Python function to check if a given number is an Armstrong number.
- Answer:

```
def is_armstrong(num):
    order = len(str(num))
    sum = 0
    temp = num
    while temp > 0:
        digit = temp % 10
        sum += digit ** order
        temp //= 10
    return num == sum
```

5. **Find the Intersection of Two Arrays:**

- Question: Write a Python function to find the intersection of two arrays.
- Answer:

```
def intersection(nums1, nums2):
    return list(set(nums1) & set(nums2))
```

6. **Find Second Largest Element in a List:**
    - Question: Write a Python function to find the second largest element in a given list.
    - Answer:

```
def second_largest(lst):
    if len(lst) < 2:
        return None
    lst.sort()
    return lst[-2]
```

7. **Calculate Fibonacci Series:**
    - Question: Write a Python function to generate the Fibonacci series up to a specified limit.
    - Answer:

```
def fibonacci(n):
    fib = [0, 1]
    while fib[-1] + fib[-2] <= n:
        fib.append(fib[-1] + fib[-2])
    return fib
```

These additional questions cover various topics such as linked lists, Armstrong numbers, Fibonacci series, and more, showcasing different problem-solving techniques in Python.

Here are some more Python problem-solving interview questions along with their answers:

1. **Find the Longest Common Prefix:**
    - Question: Write a Python function to find the longest common prefix string amongst a list of strings.
    - Answer:

    ```
    def longest_common_prefix(strs):
        if not strs:
            return ""
        min_str = min(strs, key=len)
        for i, char in enumerate(min_str):
            for string in strs:
                if string[i] != char:
                    return min_str[:i]
        return min_str
    ```

2. **Rotate Array:**
    - Question: Write a Python function to rotate an array to the right by k steps.
    - Answer:

    ```
    def rotate_array(nums, k):
        k %= len(nums)
        nums[:] = nums[-k:] + nums[:-k]
    ```

3. **Merge Two Sorted Lists:**
    - Question: Write a Python function to merge two sorted linked lists and return it as a new sorted list.
    - Answer:

    ```
    class ListNode:
        def __init__(self, val=0, next=None):
            self.val = val
            self.next = next

    def merge_sorted_lists(l1, l2):
        dummy = ListNode()
    ```

```
    current = dummy
    while l1 and l2:
        if l1.val < l2.val:
            current.next = l1
            l1 = l1.next
        else:
            current.next = l2
            l2 = l2.next
        current = current.next
    current.next = l1 or l2
    return dummy.next
```

4. **Find All Anagrams in a String:**
    - Question: Write a Python function to find all the start indices of the anagrams of a given string in another string.
    - Answer:

```
def find_anagrams(s, p):
    result = []
    p_count = collections.Counter(p)
    s_count = collections.Counter(s[:len(p)-1])
    for i in range(len(p)-1, len(s)):
        s_count[s[i]] += 1
        if s_count == p_count:
            result.append(i-len(p)+1)
        s_count[s[i-len(p)+1]] -= 1
        if s_count[s[i-len(p)+1]] == 0:
            del s_count[s[i-len(p)+1]]
    return result
```

5. **Find the First Non-Repeating Character in a String:**
    - Question: Write a Python function to find the first non-repeating character in a string.
    - Answer:

```
def first_non_repeating_char(s):
    count = collections.Counter(s)
    for char in s:
        if count[char] == 1:
            return char
    return ''
```

These additional questions cover various data structures and algorithms, including strings, arrays, and linked lists, providing a comprehensive overview of problem-solving skills in Python.

String :

Certainly! Here are some Python string-related problem-solving interview questions along with their answers:

1. **Reverse Words in a String:**
   - Question: Write a Python function to reverse words in a given string.
   - Answer:

   ```
   def reverse_words(s):
       return ' '.join(s.split()[::-1])
   ```

2. **Check for Substring:**
   - Question: Write a Python function to check if a given substring exists in a given string.
   - Answer:

   ```
   def is_substring(s, substring):
       return substring in s
   ```

3. **Count Vowels and Consonants:**
   - Question: Write a Python function to count the number of vowels and consonants in a given string.
   - Answer:

```
def count_vowels_consonants(s):
    vowels = 'aeiouAEIOU'
    vowel_count = sum(1 for char in s if char in vowe
ls)
    consonant_count = len(s) - vowel_count
    return vowel_count, consonant_count
```

4. **Check for Pangram:**
    - Question: Write a Python function to check if a given string is a pangram (contains every letter of the alphabet at least once).
    - Answer:

```
import string

def is_pangram(s):
    return set(string.ascii_lowercase) <= set(s.lower
())
```

5. **Remove Duplicates from String:**
    - Question: Write a Python function to remove duplicates characters from a given string while maintaining the order of characters.
    - Answer:

```
def remove_duplicates(s):
    seen = set()
    return ''.join(char for char in s if not (char in
seen or seen.add(char)))
```

6. **Check for Anagrams:**
    - Question: Write a Python function to check if two strings are anagrams of each other.
    - Answer:

```
def are_anagrams(str1, str2):
    return sorted(str1) == sorted(str2)
```

7. **String Compression:**
   - Question: Write a Python function to perform basic string compression using the counts of repeated characters.
   - Answer:

```
def compress_string(s):
    result = []
    count = 1
    for i in range(1, len(s)):
        if s[i] == s[i - 1]:
            count += 1
        else:
            result.append(s[i - 1] + str(count))
            count = 1
    result.append(s[-1] + str(count))
    compressed = ''.join(result)
    return compressed if len(compressed) < len(s) els
e s
```

These questions focus specifically on manipulating and processing strings in Python, showcasing various techniques and methods used in string manipulation.

List:

Here are some Python problem-solving interview questions related to lists along with their answers:

1. **Find Maximum Subarray Sum:**
   - Question: Write a Python function to find the maximum sum of a contiguous subarray within a given list of integers.
   - Answer:

```
def max_subarray_sum(nums):
    max_sum = current_sum = nums[0]
    for num in nums[1:]:
        current_sum = max(num, current_sum + num)
        max_sum = max(max_sum, current_sum)
    return max_sum
```

2. **Remove Duplicates from Sorted List:**
    - Question: Write a Python function to remove duplicates from a sorted list.
    - Answer:

```
def remove_duplicates_sorted(nums):
    if not nums:
        return 0
    i = 0
    for j in range(1, len(nums)):
        if nums[j] != nums[i]:
            i += 1
            nums[i] = nums[j]
    return i + 1
```

3. **Rotate List:**
    - Question: Write a Python function to rotate a given list to the right by k steps.
    - Answer:

```
def rotate_list(nums, k):
    k %= len(nums)
    nums[:] = nums[-k:] + nums[:-k]
```

4. **Merge Sorted Lists:**
    - Question: Write a Python function to merge two sorted lists into a single sorted list.
    - Answer:

```python
def merge_sorted_lists(nums1, nums2):
    i = j = 0
    merged = []
    while i < len(nums1) and j < len(nums2):
        if nums1[i] < nums2[j]:
            merged.append(nums1[i])
            i += 1
        else:
            merged.append(nums2[j])
            j += 1
    merged.extend(nums1[i:])
    merged.extend(nums2[j:])
    return merged
```

5. **Find Single Number:**
    - Question: Write a Python function to find the single number in a non-empty list where every element appears twice except for one.
    - Answer:

    ```python
    def single_number(nums):
        result = 0
        for num in nums:
            result ^= num
        return result
    ```

6. **Find Intersection of Two Lists:**
    - Question: Write a Python function to find the intersection of two lists.
    - Answer:

    ```python
    def intersection(nums1, nums2):
        return list(set(nums1) & set(nums2))
    ```

These questions focus on various operations and manipulations that can be performed on lists in Python, including finding maximum subarray sum, removing duplicates, rotating lists, merging sorted lists, and more.

set:

Here are some Python problem-solving interview questions related to sets along with their answers:

1. **Find Intersection of Two Sets:**
    - Question: Write a Python function to find the intersection of two sets.
    - Answer:

    ```
    def intersection(set1, set2):
        return set1.intersection(set2)
    ```

2. **Find Union of Two Sets:**
    - Question: Write a Python function to find the union of two sets.
    - Answer:

    ```
    def union(set1, set2):
        return set1.union(set2)
    ```

3. **Find Difference of Two Sets:**
    - Question: Write a Python function to find the difference between two sets.
    - Answer:

    ```
    def difference(set1, set2):
        return set1.difference(set2)
    ```

4. **Check if Subset:**
    - Question: Write a Python function to check if one set is a subset of another.
    - Answer:

    ```
    def is_subset(set1, set2):
        return set1.issubset(set2)
    ```

5. **Check if Superset:**
   - Question: Write a Python function to check if one set is a superset of another.
   - Answer:

   ```
   def is_superset(set1, set2):
       return set1.issuperset(set2)
   ```

6. **Remove Duplicates from List using Set:**
   - Question: Write a Python function to remove duplicates from a list using a set.
   - Answer:

   ```
   def remove_duplicates(lst):
       return list(set(lst))
   ```

7. **Find Symmetric Difference:**
   - Question: Write a Python function to find the symmetric difference between two sets.
   - Answer:

   ```
   def symmetric_difference(set1, set2):
       return set1.symmetric_difference(set2)
   ```

8. **Count Distinct Elements in List using Set:**
   - Question: Write a Python function to count the number of distinct elements in a list using a set.
   - Answer:

   ```
   def count_distinct_elements(lst):
       return len(set(lst))
   ```

These questions focus on various set operations such as finding intersection, union, difference, checking for subset and superset, removing duplicates from a list, finding symmetric difference, and counting distinct elements in a list using sets in Python.

dictionaries:

Here are some Python problem-solving interview questions related to dictionaries along with their answers:

1. **Check Key Existence:**
    - Question: Write a Python function to check if a key exists in a dictionary.
    - Answer:

    ```
    def key_exists(dictionary, key):
        return key in dictionary
    ```

2. **Access Value by Key:**
    - Question: Write a Python function to access the value associated with a key in a dictionary.
    - Answer:

    ```
    def get_value(dictionary, key):
        return dictionary.get(key)
    ```

3. **Merge Two Dictionaries:**
    - Question: Write a Python function to merge two dictionaries.
    - Answer:

    ```
    def merge_dicts(dict1, dict2):
        return {**dict1, **dict2}
    ```

4. **Find Common Keys:**
    - Question: Write a Python function to find common keys between two dictionaries.
    - Answer:

    ```
    def common_keys(dict1, dict2):
        return dict1.keys() & dict2.keys()
    ```

5. **Find Unique Values:**
    - Question: Write a Python function to find unique values in a dictionary.
    - Answer:

    ```
    def unique_values(dictionary):
        return set(dictionary.values())
    ```

6. **Check Anagram Pairs:**
    - Question: Write a Python function to check if two strings are anagram pairs present as values in a dictionary.
    - Answer:

    ```
    def are_anagram_pairs(dictionary, str1, str2):
        return sorted(dictionary.get(str1, '')) == sorted
    (dictionary.get(str2, ''))
    ```

7. **Remove Key-Value Pair:**
    - Question: Write a Python function to remove a key-value pair from a dictionary.
    - Answer:

    ```
    def remove_key(dictionary, key):
        if key in dictionary:
            del dictionary[key]
        return dictionary
    ```

8. **Count Occurrences of Characters:**

- Question: Write a Python function to count occurrences of characters in a string and store them in a dictionary.
- Answer:

  ```
  def count_characters(string):
      char_count = {}
      for char in string:
          char_count[char] = char_count.get(char, 0) +
  1
      return char_count
  ```

These questions focus on various operations and manipulations that can be performed on dictionaries in Python, including checking key existence, accessing values, merging dictionaries, finding common keys, finding unique values, checking anagram pairs, removing key-value pairs, and counting occurrences of characters.